**1. Overload a Function to Concatenate Two Strings and Two Character Arrays Separately**

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

string concatenate(string a, string b) {

return a + b;

}

char\* concatenate(const char\* a, const char\* b) {

char\* result = new char[strlen(a) + strlen(b) + 1];

strcpy(result, a);

strcat(result, b);

return result;

}

int main() {

string str1 = "Hello, ";

string str2 = "World!";

cout << "String Concatenation: " << concatenate(str1, str2) << endl;

const char\* charArr1 = "Good ";

const char\* charArr2 = "Morning!";

char\* result = concatenate(charArr1, charArr2);

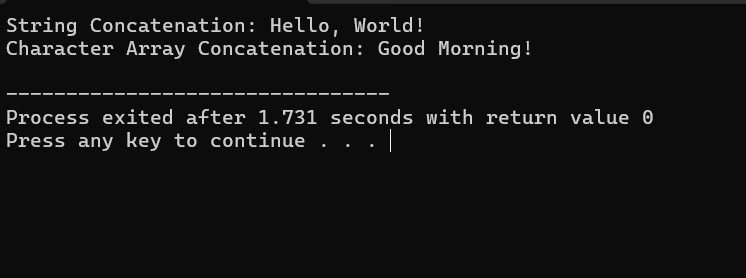
cout << "Character Array Concatenation: " << result << endl;

delete[] result;

return 0;

}

**Output**

****

**2. Overload a Function to Calculate the Sum of Two Matrices and Two Arrays Separately**

#include <iostream>

using namespace std;

void sum(int a[], int b[], int result[], int size) {

for (int i = 0; i < size; i++) {

result[i] = a[i] + b[i];

}

}

void sum(int a[2][2], int b[2][2], int result[2][2]) {

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

result[i][j] = a[i][j] + b[i][j];

}

}

}

int main() {

int arr1[3] = {1, 2, 3};

int arr2[3] = {4, 5, 6};

int arrResult[3];

sum(arr1, arr2, arrResult, 3);

cout << "Array Sum: ";

for (int i = 0; i < 3; i++) {

cout << arrResult[i] << " ";

}

cout << endl;

int mat1[2][2] = {{1, 2}, {3, 4}};

int mat2[2][2] = {{5, 6}, {7, 8}};

int matResult[2][2];

sum(mat1, mat2, matResult);

cout << "Matrix Sum: " << endl;

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

cout << matResult[i][j] << " ";

}

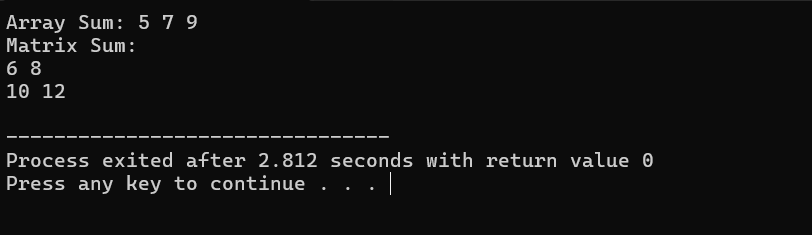
cout << endl;

}

return 0;

}

**Output**

****

**3. Overload a Function to Find Factorial of an Integer Number and Floating-Point Number Separately**

#include <iostream>

using namespace std;

int factorial(int n) {

if (n <= 1) return 1;

return n \* factorial(n - 1);

}

double factorial(double n) {

if (n <= 1.0) return 1.0;

return n \* factorial(n - 1.0);

}

int main() {

int intVal = 5;

double doubleVal = 5.0;

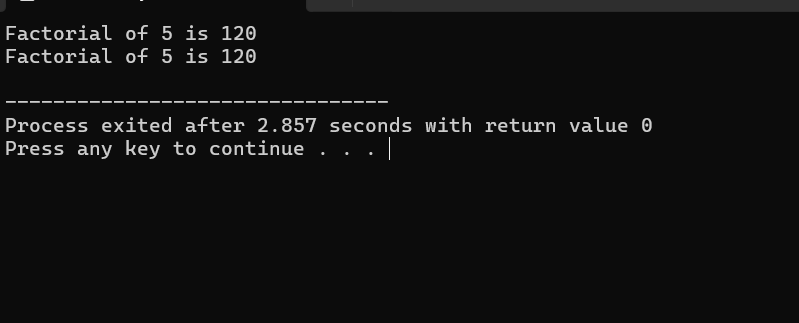
cout << "Factorial of " << intVal << " is " << factorial(intVal) << endl;

cout << "Factorial of " << doubleVal << " is " << factorial(doubleVal) << endl;

return 0;

}

**Output**

****

**4. Overload a Function to Calculate the Power of an Integer Number and Floating-Point Number Separately**

#include <iostream>

#include <cmath>

using namespace std;

int power(int base, int exponent) {

return pow(base, exponent);

}

double power(double base, double exponent) {

return pow(base, exponent);

}

int main() {

int intBase = 2;

int intExponent = 3;

double doubleBase = 2.5;

double doubleExponent = 3.0;

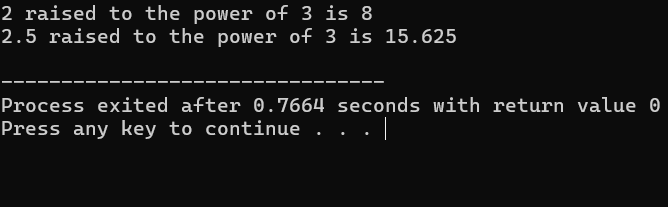
cout << intBase << " raised to the power of " << intExponent << " is " << power(intBase, intExponent) << endl;

cout << doubleBase << " raised to the power of " << doubleExponent << " is " << power(doubleBase, doubleExponent) << endl;

return 0;

}

**Output**

****

**5. Overload a Function to Find Absolute Value of an Integer Number and Floating-Point Number Separately**

#include <iostream>

#include <cmath>

using namespace std;

int absolute(int n) {

return abs(n);

}

double absolute(double n) {

return fabs(n);

}

int main() {

int intVal = -5;

double doubleVal = -5.5;

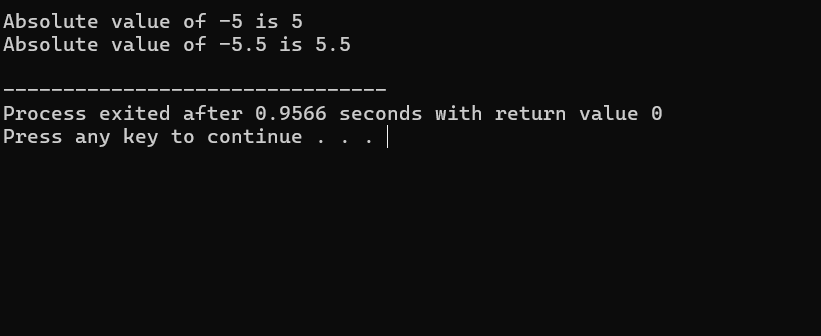
cout << "Absolute value of " << intVal << " is " << absolute(intVal) << endl;

cout << "Absolute value of " << doubleVal << " is " << absolute(doubleVal) << endl;

return 0;

}

**Output**

****

**14. Overload a Function to Print an Integer Array, a Double Array, and a Character Array Separately**

#include <iostream>

using namespace std;

void printArray(int arr[], int size) {

for (int i = 0; i < size; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

void printArray(double arr[], int size) {

for (int i = 0; i < size; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

void printArray(char arr[]) {

for (int i = 0; arr[i] != '\0'; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

int main() {

int intArr[] = {1, 2, 3, 4, 5};

double doubleArr[] = {1.1, 2.2, 3.3, 4.4, 5.5};

char charArr[] = "Hello";

cout << "Integer Array: ";

printArray(intArr, 5);

cout << "Double Array: ";

printArray(doubleArr, 5);

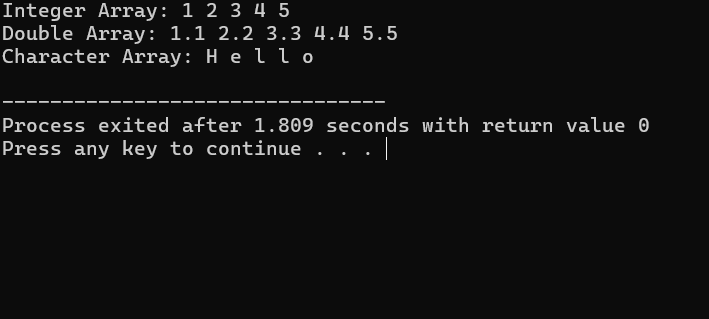
cout << "Character Array: ";

printArray(charArr);

return 0;

}

**Output**

****

**16. Overload a Function to Sort an Integer Array and a Double Array**

#include <iostream>

#include <algorithm>

using namespace std;

void sortArray(int arr[], int size) {

sort(arr, arr + size);

}

void sortArray(double arr[], int size) {

sort(arr, arr + size);

}

int main() {

int intArr[] = {5, 3, 4, 1, 2};

double doubleArr[] = {5.5, 3.3, 4.4, 1.1, 2.2};

cout << "Original Integer Array: ";

for (int i = 0; i < 5; i++) {

cout << intArr[i] << " ";

}

cout << endl;

cout << "Original Double Array: ";

for (int i = 0; i < 5; i++) {

cout << doubleArr[i] << " ";

}

cout << endl;

sortArray(intArr, 5);

sortArray(doubleArr, 5);

cout << "Sorted Integer Array: ";

for (int i = 0; i < 5; i++) {

cout << intArr[i] << " ";

}

cout << endl;

cout << "Sorted Double Array: ";

for (int i = 0; i < 5; i++) {

cout << doubleArr[i] << " ";

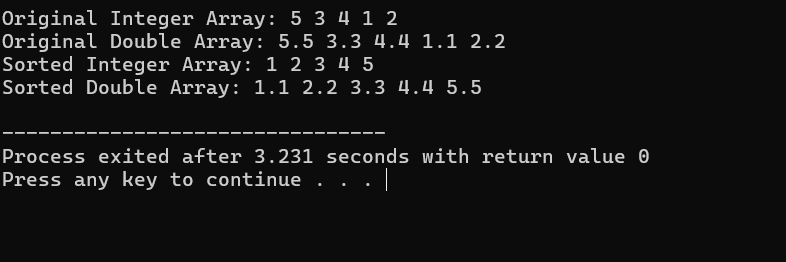
}

cout << endl;

return 0;

}

**Output**

****